НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Автоматизованих систем обробки інформації і управління

«На правах рукопису»
УДК 004.67 .

«До захисту допущено»
Завідувач кафедри
_____ О.А.Павлов
«___»_____2018 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 121 Інженерія програмного забезпечення

на тему: «Застосування графових баз даних для управління частково структурованими даними у системі управління контентом»

Виконав (-ла):
студент (-ка) VI курсу, групи ІП-61м
Носов Костянтин Сергійович _____

Керівник:
Доцент
Муха Ірина Павлівна. _____

Консультант:
Доцент, кандидат технічних наук
_____

Рецензент:
_____
_____ _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2018 року

Реферат

Магістерська дисертація:

51 сторінок, 13 рисунків, 2 таблиць, 3 додатка, 24 джерел

Актуальність теми магістерського дослідження: на сьогодні найбільш популярні системи управління контентом (CMS - content managment system) не враховують виклики сьогодення. Більшість систем побудовані на базі класичних реляційних баз даних і мають вади, які полягають у порівняно низькій продуктивності при роботі з великими об’ємами даних або при роботі з даними, які мають складну мережеву структуру (мають велику кількість зав’язків між собою).

Тому задача структуризації великих об’ємів даних, а також даних, які мають складну мережеву структуру, в системах управління контентом є достатньо актуальною.

Робота присвячена розробці методів структуризації даних для управління частково структурованим контентом.

Мета дослідження. Метою магістерської роботи є дослідження існуючих підходів до управління даними та удосконалення методів структуризації даних при вирішенні задач управління частково структурованим контентом для підвищення продуктивності роботи систем управління контентом при роботі з даними, які мають складну мережеву структуру.

Завдання дослідження. Для досягнення поставленої мети необхідно вирішити наступні задачі:

- дослідити існуючі методи управління при роботі з великими обсягами частково структурованого контенту;
- дослідити існуючі моделі баз даних для управління частково структурованим контентом;
- адаптувати графічну модель баз даних для управління складно структурованим контентом;

– розробити систему управління контентом з реалізацією запропонованого методу структуризації даних.

Об'єктом дослідження є процеси управління частково структурованим контентом та процес структурування контенту.

Предметом дослідження є моделі репрезентації частково структурованих даних як основа проектування і реалізації системи управління контентом.

Методи дослідження. Теорія графів, теорія баз даних, сучасні моделі дослідження і репрезентації даних.

Наукова новизна отриманих результатів полягає в розробці та реалізації методу структуризації даних, які мають складну мережеву структуру, за допомогою використання графових баз даних і динамічного створення власних типів даних, що підвищує структуризацію контенту і, як наслідок, прискорює пошук і вибірку даних.

Практична значимість отриманих результатів. В результаті виконаного магістерського дослідження запропонована модифікована модель графічної репрезентації даних. На основі запропонованої моделі розроблена і реалізована програмна система на базі якої були імплементовані наступні проекти - https://coin.ua, https://homeinvesting.ru

Публікації. Матеріали роботи опубліковані в тезах конференцій «ІНФОРМАТИКА ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА – ІОТ-2018» та «XIX Міжнародна конференція за математичного моделювання – МКММ-18» (Херсон, ХНТУ, 2018)

Abstract

Master's dissertation:

51 page, 13 images, 2 tables, 3 additions, 24 sources

Relevance of the topic of the master's study: today the most popular content management systems (CMS) do not take into account the challenges of the present. Most systems are based on classic relational databases and have weaknesses that are relatively low in performance when working with large volumes of data or when working with data that has a complex network structure (with a large number of ties among themselves).

Therefore, the task of structuring large volumes of data, as well as data that have a complex network structure, in content management systems is quite relevant.

The work is devoted to the development of data structuring methods for the management of partially structured content.

The aim of the study. The purpose of the master's thesis is to study existing data management approaches and to improve data structuring methods when solving the problems of partially structured content management to improve the performance of content management systems while working with data that has a complex network structure.

Objectives of the study. To achieve this goal, the following tasks must be solved:

- to investigate existing management methods when dealing with large volumes of partially structured content;
- explore existing database models for managing partially structured content;
- adapt a graphical database model to manage complexly structured content;
- develop a content management system with the implementation of the proposed method of structuring data.

The object of the study is the processes of managing partially structured content and the process of structuring content.

The subject of the study is the models for the representation of partially structured data as the basis for the design and implementation of a content management system.

Research methods. Theory of graphs, database theory, modern models of research and data representation.

The scientific novelty of the results obtained is to develop and implement a method for structuring data that has a complex network structure, using graph databases and dynamically creating their own data types, which increases the structuring of the content and, as a result, accelerates the search and selection of data.

The practical significance of the results. As a result of the completed master's study, a modified model of graphical representation of data is proposed. On the basis of the proposed model, a program system was developed and implemented on the basis of which the following projects were implemented: https://coin.ua, https://homeinvesting.ru

Publications Materials of work are published in the theses of conferences "INFORMATICS AND COMPUTER TECHNOLOGY - IOT-2018" and "XIX International Conference on Mathematical Modeling - MKMM-18" (Kherson, KhNTU, 2018)

Умовні позначення

- CMS – Conten Management System – система управління контентом для редакторів сайту, для спрощення
- WP – WordPress, поширена CMS,
- Контент – інформаційне наповнення сайту
- Веб застосунки – застосунки призначенні для роботи в мережі інтенрнет за допомогою http протоколу, створенні у HTML за допомогою CSS та JavaScript

Зміст

ВСТУП:

Основна ідея систем управління контентом (CMS – Content Managing System) - розділення візуального оформлення сайту та його інформаційного наповнення. При побудові сайту за допомого, такої системи розробляється набір шаблонів сторінок, в котрих потім розмішується інформація. В цьому випадку завдання розробників (фактично це група впровадження) обмежується тільки побудовою «стартової» інформаційної системи, на основі системи управління контентом. Після цього користувачі самі публікують необхідну інформацію і вибирають її представлення. Управління сайтом для адміністратора сходить до управління користувачами.

Сутність у CMS це визначений вид контенету з додатковими полями, настройками і свойствами. Наприклад «Новина» чи «Автор» це приклади сутностей. У кожної сутності можуть бути екземпляри, до прикладу – Автор «Петя» чи новина «Тунгуський метеорит». При рості кількості сутностей насатє деградація продуктивності, аж до відмови сайту.

Для сайтів, не блогів (у блогів лінійний індекс і всього декілька типів даних, вони за своєю суттю не мають цієї проблеми), а для інформаційних порталів це складна проблема. В інформаційному порталі повинно бути багато базових сутностей, не тільки автор та стаття, но і таксономія та біль спеціалізовані типи привязані до контенту (біржа, валюта, брокер)

Якщо CMS з початку не розроблялась як адаптована для навантажень і оперування великою кількістю даних, то на певному етапі життєвого циклу вона стане занадто повільною для кінцевого користувача.

Тому задача оптимізації збереження частково структурованих данних є актуальною задачею, яка дозволяє поліпшити функціонування веб застосунків з великою кілкістю контенту.

Дана робота присвячена модифікації існуючих подходів до збереження данних в рамках CMS.

Метою є удосконалення існуючих методів зберігання та обробки частично структурованого контенту на основі графових систем для прискорення їх роботи. Для досягнення мети необхідно вирішити наступні завдання:

- дослідити проблему;
- дослідити підходи до рішення;
- запропонувати як структурувати контент;
- запропонувати спосіб зберігання контенту;
- побудувати CMS.

1 ОГЛЯД ПІДХОДІВ ЗБЕРЕЖЕННЯ І ОБРОБКИ ДАНИХ У CONTENT MANAGEMENT SYSTEMS

На даний момент є багато різноманітних систем управління контентом. Кожен інструмент відповідає тієї задачі задля якої його зроблено. В області програмного забезпечення цілі і задачі змінюються достатньо швидко, що навіть базові постулати можуть застаріти за декілька років. Можливо саме це і сталося з системами управління контентом. Дуже довгий час ми бачимо одній й ті самі парадигми, ті ж самі патерни і підходи до побудови систем управління контентом. Це реляційні бази, wysiwyg , блогова чи тегова структура сайту.

Ми можем застосувати графові бази для вирішення задачі збереження частково структурованого контенту, але спочатку подивимось які взагалі є підходи до збереження і обробки даних у CMS

На **Error! Reference source not found.**Рисунок 1.1 можна побачити найбільш вживані Системи управління контентом. З дуже великим відривом ми бачимо Wordpress попереду. Розберемо приклад використання CMS Wordpress та фреймворку.

#	WEBSITES USING	MARKET SHARE %	ACTIVE SITES	# OF WEBSITES IN MILLION
1	WordPress	59.9 %	26,701,222	239,139
2	Joomla	6.6 %	2,009,717	13,480
3	Drupal	4.6 %	964,820	23,330
4	Magento	2.4 %	372,915	12,095
5	Blogger	1.9 %	758,571	15,779
6	Shopify	1.8 %	605,506	11,587
7	Bitrix	1.5 %	200,210	3,925
8	TYPO3	1.5 %	582,629	3,568
9	Squarespace	1.5 %	1,390,307	9,799
10	PrestaShop	1.3 %	262,342	2,099

Рисунок 1.1 - Популярність використання CMS у мережі інтернет

CMS призначенні для управління частково структурованим контентом. У чому різниця між структурованим і неструктурованих контентом для сайту? Структурний контент лягає на шаблон, а неструктурований вбудовано у шаблон. Зміна дизайну або верстки ламає цей контент. CMS повинна дозволяти автору

бути десь посередні – використовувати поля і типи даних для структурованої інформації, та неструктуровано зберігати сам контент статей (структуруючи наприклад гіперпосилання та малюнки у контенті за допомогою медіа бібліотеки тощо)

1.1 Wordpress

Ця CMS є дуже старим, але постійно розвиваючимся та оновлюємим програмним продуктом. Однією з переваг її використання є наявність великої інфраструктури готових підключаємих модулів, реалізуючих різноманітні задачі, починаючи від додавання блоків з тексом, закінчуючи реалізацією багатомовності для сайту чи перетворення сайту на інтернет магазин з інтеграцією з бухгалтерським програмним забезпеченням. Це досягається наявністю у Wordpress програмних інтерфейсів і способів створення модулів званих **плагінами**, а також декількох таблиць у БД - **terms, taxonomy, term_relationships** та **term_meta** див. Рисунок 1.2. По суті це єдиний спосіб зберегти і зв'язати дані які не вписуються в стандартну модель основних таблиць Wordpress. вся суть який зводиться до початкового задуму використовувати Wordpress як cms для блогу

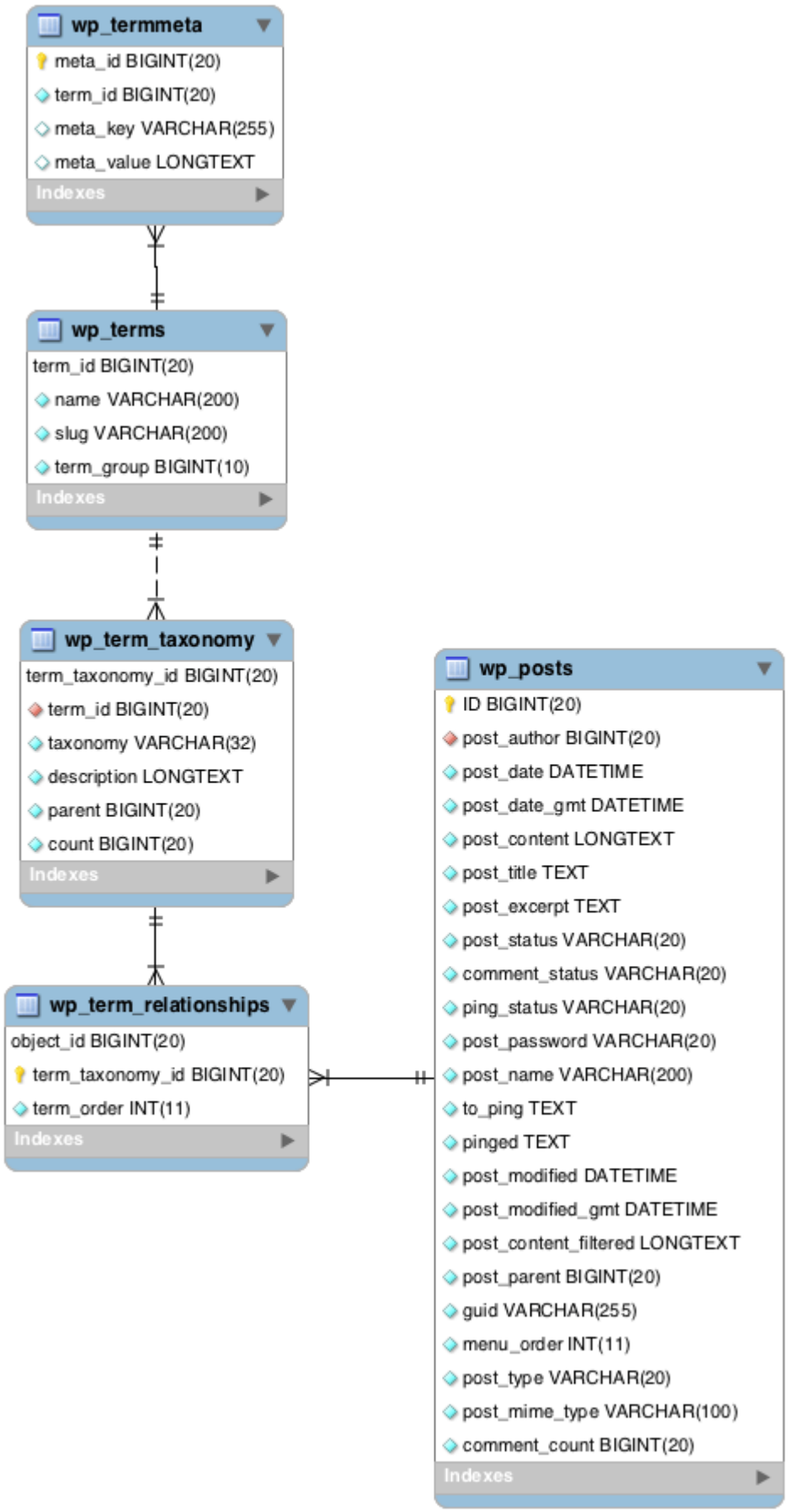

Рисунок 1.2 - ER-діаграма частини таблиць бази даних CMS Wordpress

Wordpress як ніяка інша cms схильна до проблеми масштабування - у міру того як сайт наповнюється контентом, стає старішим а значить об'ємним, зростає кількість зв'язків між його одиницями контенту. Всі вони зберігаються в таблиці Posts або в розрізнених вигляді в чотирьох вищезазначених таблицях. У моменти коли потрібно використовувати terms або taxonomy, cms може як зробити новий запит так і взяти готове значення з кешу. Що може привести до істотного уповільнення роботи сайту. Також в більшості випадків, автори сторонніх плагінів зберігають найрізноманітнішу інформацію саме в цих таблицях.

meta_id	term_id	meta_key	meta_value
1 655	2 713	content_review	
1 672	2 852	content_review	a:2:{i:0;s:9:"text_area";i:1;s:13:"youtube_video";}
1 673	2 852	content_review_0_text	Сотни заведений вовлекают новы…
1 674	2 852	tax_main_content	
2 377	12 108	_content_review	field_53a8b271c8dee
2 378	12 108	_tax_main_content	field_55e4732325c5e
2 379	4 256	_content_review	field_53a8b271c8dee
2 380	4 256	_content_review_0_text	field_53a8b2a1c8def
2 381	4 256	_tax_main_content	field_55e4732325c5e
2 382	4 257	_content_review	field_53a8b271c8dee
2 383	4 257	_tax_main_content	field_55e4732325c5e
2 384	4 258	_content_review	field_53a8b271c8dee
2 385	4 258	_tax_main_content	field_55e4732325c5e
2 386	4 259	_content_review	field_53a8b271c8dee
2 387	4 259	_content_review_0_text	field_53a8b2a1c8def
2 388	4 259	_tax_main_content	field_55e4732325c5e
2 389	4 260	_content_review	field_53a8b271c8dee

Рисунок 1.3 - реальний приклад даних зберігаємих у таблиці term_meta

Як видно з Рисунок 1.3 в поле meta_value зберігається інформація самого різного виду - серіалізований масив з мета-описом, html, id поля в текстовому вигляді. Стає очевидно що робити одну мультифункціональну таблицю для зберігання різних за форматом даних є архітектурно невірним рішенням і порушенням 1-ї Нормальною Форми в декількох випадках до ряду. У випадках коли потрібно організувати повнотекстовий пошук по даному полю, розробник ставиться в глухий кут, тому що: по-перше неможливо передбачити результати такого пошуку - частина даних задовольняють запит пошуку може зберігатися всередині серіалізованих об'єкта або масиву, по-друге тому що частина даних цієї таблиці по своїй суті зберігає ідентифікатори у вигляді рядків, швидкість виконання запитів падає катастрофічно.

1.2 Фреймворк як база для CMS

В інших cms більш поширений спосіб зберігання пов'язаних даних в різних таблицях і з використанням проміжної таблиці в разі зв'язку багато-до-багатьох. Хоча такий спосіб зв'язки даних добре себе зарекомендував і широко використовується, він також має недоліки:

- Необхідність створювати проміжні таблиці для кожної зв'язку "багато-до-багатьох";
- Необхідність адаптувати структуру даних під універсальну таблицю (як у випадку з wordpress), або ж створювати нову таблицю для зберігання даних і таблицю для зберігання зв'язків для кожного нового типу контенту.

Через такі обмеження при розробці виникає необхідність використовувати "join" - операції або користуватися результатами підзапитів в основному запиті для виведення часто запитуваної інформації кінцевому користувачу.

1.3 Постановка задач дослідження

Більшість систем управління контентом побудовані на базі класичних реляційних баз даних і мають порівняно низьку продуктивність при роботі з великими об’ємами даних, які мають складну мережеву структуру. Тому актуальною є задача розробки нових методів структуризації даних для управління частково структурованим контентом.

Метою магістерської роботи є дослідження існуючих підходів до управління даними та удосконалення методів структуризації даних при вирішенні задач управління частково структурованим контентом для підвищення продуктивності роботи систем управління контентом при роботі з даними, які мають складну мережеву структуру.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

- дослідити існуючі методи управління при роботі з великими обсягами частково структурованого контенту;

- дослідити існуючі моделі баз даних для управління частково структурованим контентом;
- адаптувати графічну модель баз даних для управління складно структурованим контентом;
- розробити систему управління контентом з реалізацією запропонованого методу структуризації даних.

Висновки

Чим більше різних типів даних ураховується у CMS, тим складнішою є система, що проектується, і тим накладніше маніпулювати даними із-за складної структуру БД з множинними зв'язками між таблицями (даними).

Виконаний аналіз існуючих систем управління контентом з точки зору реалізації структур баз даних і задач управління великими обсягами частково структурованого веб-контенту. Показано, що із зростанням кількості статей реляційна модель організації даних, загальноуживана у сучасних CMS, зокрема, у WordPress, призводить до уповільнення операцій пошуку і виборки по даним. Шляхом вирішення даної проблеми може бути використання іншої моделі організації даних, зокрема, уникнення необхідності побудови JOIN-ів для реалізації зв’язків між типами даних і самими даними.

2 МОДЕЛІ РЕПРЕЗЕНТАЦІЇ ЧАСТКОВО СТРУКТУРОВАНИХ ДАНИХ У БАЗАХ ДАНИХ

2.1 Ієрархічна модель

У ієрархічній моделі данні організовані у вигляді деревовидної структури, з огляду на те ще у кожного запису є один батько. Поле сортування зберігає записи одного рівня в визначеному порядку. Ієрархічні системи були багатовживаними в ранніх системах управління базами даних ще на мейнфреймах, як то IBM IMS, і зараз описують структуру XML документу. Ця структура дозволяє одну взаємовідносину між двома типами даних. Ця структура дуже ефективна для опису багатьох взаємозв’язків у реальному житті. Наприклад рецепти, зміст, порядок абзаців\віршів або будь яка інша вкладена та посортована інформація.

Ця ієрархія використовується як фізичний порядок записів у сховищі. Доступ до запису забезпечується шляхом навігації вниз по структурі даних за допомогою покажчиків в поєднанні з послідовним доступом. Через це ієрархічна структура неефективна для деяких операцій з базою даних, коли повний шлях (на відміну від висхідній лінії зв'язку і поля сортування) також не включений для кожного запису. Такі обмеження були компенсовані в більш пізніх версіях IMS додатковими логічними ієрархіями, накладеними на базову фізичну ієрархію.

2.2 Мережева модель

Мережева модель розширює ієрархічну структуру, дозволяючи відносини «багато до багатьох». Модель була найбільш популярною заміною для реляційної моделі і визначалася специфікацією CODASYL.

Мережева модель організовує дані з використанням двох фундаментальних понять, які називаються записами і наборами. Записи містять поля (які можуть бути організовані ієрархічно, як у мові програмування COBOL). Набори (не плутати з математичними наборами) визначають відносини «один до

багатьох» між записами: один власник, багато членів. Запис може бути власником в будь-якій кількості наборів і членом в будь-якій кількості наборів.

Набір складається з кругових пов'язаних списків, де один тип запису, власник набору або батьківський елемент, з'являється один раз в кожному колі, а другий тип запису, підлеглий або дочірній, може з'являтися кілька разів в кожному колі. Таким чином, між будь-якими двома типами записів може бути встановлена ієрархія, наприклад, тип А є власником В. У той же час може бути визначений інший набір, де В є власником А. Таким чином, усі множини містять загальний орієнтований граф (Право власності визначає напрямок) або мережеву конструкцію. Доступ до записів є або послідовним (зазвичай в кожному типі записи), або навігацією в кругових пов'язаних списках.

Мережева модель здатна представляти надмірність даних більш ефективно, ніж в ієрархічній моделі, і може бути більш одного шляху від вузла-предка до потомка. Операції мережевий моделі є навігаційними по стилю: програма підтримує поточну позицію і переміщається з одного записи до іншого, слідуючи відносинам, в яких бере участь запис. Записи також можна знайти, вказавши значення ключа.

Хоча це не є суттєвою особливістю моделі, мережеві бази даних зазвичай реалізують встановлені відносини за допомогою покажчиків, які безпосередньо адресують розташування запису на диску. Це дає відмінну продуктивність пошуку за рахунок таких операцій, як завантаження бази даних і реорганізація.

Популярними продуктами СУБД, які його використовували, були IDMS Cincom Systems і Total ID Cullinet. IDMS отримала значну клієнтську базу; У 1980-х роках він прийняв реляційну модель і SQL на додаток до своїх оригінальним інструментів і мов. Більшість об'єктних баз даних (винайдених в 1990-х роках) використовують навігаційну концепцію для швидкої навігації по мережах об'єктів, зазвичай використовуючи ідентифікатори об'єктів як «розумні» покажчики на зв'язані об'єкти. Об'єктивність / БД, наприклад, реалізує так звані відносини один до одного, один-ко-многим, багато-до-одного і багато-

до-багатьох, які можуть перехрещувати бази даних. Багато баз даних об'єктів також підтримують SQL, поєднуючи сильні сторони обох моделей.

2.3 Інвертована файлова модель

У інвертованому файлі або інвертованому індексі вміст даних використовується як ключі в таблиці пошуку, а значення в таблиці є покажчиками на розташування кожного примірника даного елемента контенту. Це також логічна структура сучасних індексів бази даних, які можуть використовувати тільки вміст з певних стовпців в таблиці пошуку. Модель даних з інвертованими файлами може поміщати індекси в другій набір файлів поруч з існуючими файлами плоских баз даних, щоб ефективно отримувати доступ до необхідних записів в цих файлах.

Показовим для використання цієї моделі даних є ADABAS DBMS of Software AG, представлена в 1970 році. ADABAS набула значної клієнтську базу і існує і підтримується до сьогоднішнього дня. У 1980-х роках вона прийняла реляційну модель і SQL на додаток до своїх оригінальних інструментів і мов.

Документарно-орієнтована база даних Clusterpoint використовує інвертовану модель індексування для забезпечення швидкого повнотекстового пошуку об'єктів даних XML або JSON і надання можливості масштабування для великих даних. Clusterpoint має вбудований обчислювальний движок, який дозволяє виконувати комбінований SQL-запит, вільний текстовий пошук і код JavaScript прямо всередині розподіленої бази даних. Як дані, так і інвертовані індекси за допомогою масштабується масштабування і реплікації можуть бути розподілені на великій кількості серверів для підтримки мільярдів об'єктів даних в одній і тій же базі даних Clusterpoint. Мова запитів Clusterpoint JS / SQL поєднує синтаксис SQL і JavaScript з повнотекстових пошуком, де інвертований покажчик використовується для доставки результатів текстового пошуку в мілісекундах і відповідної розбивки на сторінки в веб-і мобільних додатках. Інвертований індекс бази даних бази даних Clusterpoint також підтримує програмований ранжування релевантності, що дозволяє налаштовувати

результати пошуку без додаткових зусиль з кодування. Подібно реляційних баз даних, Clusterpoint підтримує розподілені транзакції бази даних, сумісні з ACID, для забезпечення узгодженої бази даних документів, де дані інвертованого індексу негайно оновлюються разом з будь-якими оновленнями вмісту документа XML або JSON. Інвертований індекс також використовується для підтримки звітів в режимі реального часу в режимі реального часу, аналітики, деталізації та інтелектуального аналізу даних по REST API в базі даних Clusterpoint.

2.4 Реляційна модель

Реляційна модель була введена Е. Ф. Коддом в 1970 році як спосіб зробити системи управління базами даних більш незалежними від якогось конкретного додатка. Це математична модель, визначена в термінах логіки предикатів і теорії множин, а використовувані нею системи використовуються в системах мейнфреймів, пк і мікрокомп'ютерів.

Продукти, які зазвичай називаються реляційними базами даних, фактично реалізують модель, яка є лише наближенням до математичної моделі, визначеної Коддом. Три ключових терміна широко використовуються в моделях реляційних баз даних: стосунки, атрибути і домени. Стосунок - це таблиця зі стовпцями і рядками. Іменовані стовпці стосунка називаються атрибутами, а домен - це набір значень, які можуть приймати атрибути.

Основною структурою даних реляційної моделі є таблиця, в якій інформація про конкретний об'єкт (скажімо, працівника) представлена в рядках (також званих кортежами) і шпальтах. Таким чином, «ставлення» в «реляційної базі даних» відноситься до різних таблиць в базі даних; Стосунок являє собою набір кортежів. У стовпчиках перераховані різні атрибути об'єкта (наприклад, ім'я співробітника, адреса або номер телефону), а рядок є фактичним екземпляром об'єкта (конкретного співробітника), який представлений відношенням. В результаті кожен кортеж таблиці employee представляє різні атрибути одного співробітника.

Всі стосунки (і, отже, таблиці) в реляційної базі даних повинні дотримуватися деяких основних правил, які можна кваліфікувати як стосунки. По-перше, впорядкування стовпців несуттєво в таблиці. По-друге, не може бути однакових кортежів або рядків в таблиці. І, по-третє, кожен кортеж буде містити одне значення для кожного з його атрибутів.

Реляційна база даних містить кілька таблиць, кожна з яких схожа на таку в «плоскій» моделі бази даних. Одна із сильних сторін реляційної моделі полягає в тому, що в принципі будь-яке значення, яке має місце в двох різних записах (що належать до однієї таблиці або до різних таблицях), має на увазі взаємозв'язок між цими двома записами. Проте, для забезпечення явних обмежень цілісності відносини між записами в таблицях також можуть бути визначені явно, шляхом ідентифікації відносин батько-дитина, що характеризуються привласненням потужності (1: 1, (0) 1: M, M: M). Таблиці також можуть мати призначений єдиний атрибут або набір атрибутів, які можуть діяти як «ключ», які можуть використовуватися для однозначної ідентифікації кожного кортежу в таблиці.

Ключ, який може використовуватися для однозначної ідентифікації рядка в таблиці, називається первинним ключем. Ключі зазвичай використовуються для об'єднання або об'єднання даних з двох або більше таблиць. Наприклад, таблиця Employee може містити стовпець з ім'ям Location, який містить значення, відповідне ключу таблиці Location. Ключі також важливі при створенні індексів, які полегшують швидкий пошук даних з великих таблиць. Будь який стовпець може бути ключем, або декілька стовпців можуть бути згруповані разом в складений ключ. Немає необхідності заздалегідь визначати всі ключі; Стовпець можна використовувати в якості ключа, навіть якщо він спочатку не був призначений для цієї мети..

Ключ, який має зовнішній, реальний сенс (наприклад, ім'я людини, ISBN книги або серійний номер автомобіля), іноді називають «природним» ключем. Якщо ніякої природний ключ не підходить (подумайте про багатьох людей на ім'я Іван), може бути призначений довільний або сурогатний ключ (наприклад,

вказуючи ідентифікаційні номери співробітників). На практиці більшість баз даних мають як згенеровані, так і природні ключі, оскільки згенеровані ключі можуть використовуватися всередині, щоб створювати зв'язку між рядками, які не можуть зламатися, в той час як природні ключі можна використовувати, менш надійно, для пошуку і для інтеграції з іншими базами даних. (Наприклад, записи в двох незалежно розроблених базах даних можуть бути співставлені номером соціального страхування, за винятком випадків, коли номери соціального страхування невірні, відсутні або змінені).

Найбільш поширеним мовою запитів, що використовуються з реляційною моделлю, є мова структурованих запитів (SQL).

2.5 Розмірна модель

Дана модель є спеціалізованою адаптацією реляційної моделі, використовуваної для представлення даних в сховищах даних, таким чином, що дані можна легко підсумувати за допомогою інтерактивної аналітичної обробки або запитів OLAP. У розмірній моделі схема бази даних складається з однієї великої таблиці фактів, які описуються з використанням вимірювань і заходів. Вимірювання надає контекст факту (наприклад, хто брав участь, коли і де це сталося, і його тип) і використовується в запитах для угруповання пов'язаних фактів. Розміри мають тенденцію бути дискретними і часто є ієрархічними; Наприклад, місце розташування може включати будівлю, штат і країну. Міра - це кількість, що описує факт, такий як дохід. Важливо, щоб заходи могли бути в значній мірі агреговані, наприклад, доходи від різних місць можуть бути об'єднані разом.

У запиті OLAP вибираються вимірювання, а факти групуються і об'єднуються разом для створення зведення.

Розмірна модель часто реалізується поверх реляційної моделі, використовуючи зоряну схему, що складається з однієї високо нормованої таблиці, яка містить факти, і оточуючих денормалізованних таблиць, що містять

кожен вимір. Альтернативна фізична реалізація, звана схемою сніжинок, нормалізує багаторівневі ієрархії всередині вимірювання в кілька таблиць.

Сховище даних може містити багатовимірні схеми, які обмінюються таблицями вимірювань, що дозволяє використовувати їх разом. Створення стандартного набору вимірювань є важливою частиною моделювання розмірів.

Його висока продуктивність зробила розмірну модель найпопулярнішою структурою бази даних для OLAP.

2.6 Пост-реляційні моделі баз даних

Продукти, що пропонують більш загальну модель даних, ніж реляційна модель, іноді класифікуються як пост-реляційні. Альтернативні терміни включають «гібридну базу даних», «RDBMS з розширеними об'єктами» та інші. Модель даних в таких продуктах включає в себе стосунки, але не обмежується інформаційним принципом Кодда, який вимагає, щоб вся інформація в базі даних була явно вказана з точки зору значень у відносинах.

Деякі з цих розширень до реляційної моделі об'єднують поняття з технологій, які передують реляційної моделі. Наприклад, вони дозволяють уявити орієнтований граф з деревами на вузлах. Німецька компанія sones реалізує цю концепцію в своєму GraphDB.

Деякі пост-реляційні продукти розширюють реляційні системи з нереляційними функціями.

2.7 Графова модель

Графові бази даних допускають ще більш загальну структуру, ніж мережева база даних; Будь-який вузол може бути підключений до будь-якого іншого вузла.

Графову модель даних зазвичай розглядають як узагальнення RDF-моделі або мережевої моделі даних (2008). Основними елементами моделі є вузли і зв'язки. Залежно від реалізації вузлів і ребер графову модель даних поділяють на кілька підтипів.

Графові бази даних застосовуються для моделювання соціальних графів (соціальних мереж) (htt), в біоінформатики, а також для семантичної павутини (Hoff, 2009). У графових СУБД, як правило, поділяють сховище (англ. Underlying storage) і механізм обробки (англ. Processing engine) (Robinson, 2013).

Для завдань з природною графовою структурою даних графові СУБД можуть значно перевищувати реляційні по продуктивності, а також мати переваги в наочності уявлення і простоті внесення змін до схеми бази даних (Robinson, 2013).

2.7.1 Механізми графових обчислень

Для аналітичної роботи з великими обсягами даних в глобальних графах застосовуються спеціалізовані механізми графових обчислень (англ. Graph compute engine). На відміну від графових СУБД, орієнтованих в основному на OLTP-додатки, в системах графових обчислень використовуються підходи і методи оптимізації, властиві OLAP. Існують різні реалізації механізмів для графових обчислень, як резидентні (англ. In-memory), так і використовують енергонезалежні пристрої зберігання, як працюють на одному вузлі, так і розподілені (працюють на декількох вузлах одночасно) (Robinson, 2013)

2.7.2 Графові системи для відсутніх даних

Відсутні дані представляють собою проблему в багатьох галузях емпіричних наук. Датчики не завжди працюють надійно, респонденти не заповнюють усі питання в анкеті, а також пацієнтів медичних установ часто не можуть згадати епізоди, процедури і результати. Статистична література з цієї проблеми багата і різноманітна, і це призвело до появи потужних пакетів програмного забезпечення, таких як MICE в R, Stata, SAS і SPSS, які пропонують різні способи подолання відсутності даних. Більшість практик ґрунтуються на основній роботі Рубіна (Rubin, 1976), який сформулював процедури та умови, за яких можна зменшити ушкодження через відсутність даних. Ця теорія також привела до ряду гарантій продуктивності, якщо дані підпорядковуються певним

статистичним умовам. Проте ці умови є досить сильними, і надзвичайно складно з'ясувати реальні проблеми. Літл і Рубін (Little, 2002), узагальнюючи сучасний стан, спостерігаючи: "по суті вся література з багатовимірних неповних даних передбачає, що дані відсутні випадково (MAR)". Дійсно, популярні методи оцінки відсутності даних, таких як методи, що базуються на максимальній вірогідності (Dempster, 1977) та Багаторазове Втручання (Rubin, 1978) вимагають прийняття MAR, щоб гарантувати конвергенцію до послідовних оцінок. Крім того, це практично неможливо для практикуючого статистика вирішити, чи відповідає умовам MP у заданій проблемі.

Протягом останніх років спостерігається все більший інтерес до аналізу відсутніх даних за допомогою графічних моделей для кодування припущень про причини відсутності. Цей розвиток є природнім, оскільки графічні моделі забезпечують ефективне представлення умовних незалежнностей, які передбачаються моделюючими припущеннями. Більш ранні роботи в цьому напрямку обговорювалися в (Daniel, 2012) який надав достатні критерії, за якими послідовні оцінки можна обчислити виключно з повних випадків (тобто зразки, в яких повністю перебувають змінні). (Thoemmes, 2013) (а пізніше (Thoemmes, 2015)) розробили методи, які спрямовують вибір допоміжних змінних для підвищення оцінки з неповних даних. У машинному навчанні, особливо при оцінюванні параметрів байєсівських мереж, графічні моделі давно використовуються як інструмент для вирішення відсутніх даних ((Darwiche, 2009), (Koller, 2009)).

У даній роботі ми розглядаємо внесок графічних моделей у відсутність досліджень даних та підкреслюємо три основні аспекти: прозорість, відновлюваність (послідовна оцінка) і Тестування

На противагу традиційному змагальному припущенню найгіршого аналізу, багато джерел даних моделюються ланцюгами Маркова (наприклад, в черзі, мові, жесті, гомології білків, веб-пошуку тощо). Ці моделі дуже привабливі, оскільки вони широко застосовуються і прості у виробництві. Дійсно, розташування посилань, важлива опора у розробці ефективних обчислювальних

систем, часто фіксується ланцюгом Маркова, що моделює розподіл доступу. Отже, не дивно, що це з'єднання мотивувало і керувало великою частиною досліджень щодо само організаційних структур даних та онлайнових алгоритмів в установці Маркова. Цей предмет роботи слід розглядати як частину більшого зусилля, щоб зрозуміти алгоритми, які використовують той факт, що розподіли вхідних даних часто демонструють лише невелику кількість ентропії. Цим зусиллям керують не тільки надії на вдосконалення практичних застосувань (наприклад, експлуатація узгодженості в потоках даних), але також обумовлені теоретичними питаннями: наприклад, ключ до вирішення проблеми проектування оптимального детерміністичного алгоритму для мінімального простягання простягання дерев, полягає у виявленні оптимальної купи джерел постійної ентропії (Chazelle, 2000). Інтенсивно вивчалися ланцюги Маркова, та існує величезна література про них (наприклад, (D. A. Levin, 2009)). Тим не менш, основна увага приділялася функціям будови (такими як стаціонарний розподіл або час переміщення/покриття/змішування), а не поведінка складних об'єктів, що розвиваються над ними. Це призводить до ряду фундаментальних питань, які, як ми сподіваємось, надихатимуть подальші дослідження.

Давайте більш докладно опишемо нашу модель. Наш об'єкт інтересу - це структура T (X), яка розвивається з часом. Структура T (X) визначена над кінцевою підмножиною X всесвіту U. У найпростішому випадку ми маємо U = N і T (X) = X.

Це відповідає класичній проблемі словника, де нам потрібно підтримувати підмножину заданого всесвіту. Також ми можемо уявити більш складні сценарії, такі як $U = R^d$, при T (X) - триангуляція Делоне по X. Граф подій G = (V, E) визначає обмеження на запити та оновлення, які застосовуються до T (X). Для простоти, ми припускаємо, що G неорієнтовано і пов'язано. Кожен вузол $v \in V$ асоційований з елементом $x_v \in U$ і відповідає одному з трьох можливих запитів: (i) insert (x_v); (ii) видалити (x_v); або (iii) запит (x_v). Запити вказуються шляхом прогулянки в G, починаючи з призначеного початкового вузла G і перескочивши від вузла до сусіднього вузла. Ми розглядаємо обидві змагальні роботи, в яких

сусідні структури можуть бути обрані довільно, а випадкові роботи, в яких сусідня структура вибирається рандомно. Останній випадок відповідає класичній моделі Маркова. Нехай v^t буде вузол G, відвіданий в момент часу t, і нехай $X^t \subseteq U$ являє собою набір активних елементів, тобто множину елементів, вставлених до часу t, а не видалених після їх останніх вставлень. Ми також називаємо X^t активним набором. Для будь-якого $t > 0$, $X^t = X^{t-1} \cup (x_v^t)$, якщо операція на v^t є вставкою, а $X^t = X^{t-1} \setminus (x_v^t)$ у випадку видалення. Запит на v залежить від розглянутої структури (наступника, точки розташування, проміння стрільби тощо). Інший спосіб інтерпретувати графік подій - як кінцевий автомат, який генерує слова по алфавіту з певними правилами скасування.

Ланцюги Маркова спираються на забування минулого. Проте в нашій моделі структура $T (X^t)$ може цілком пам'ятати. Фактично, ми можемо визначити вторинний граф на набагато більшому наборі вершин $V \times 2^{U_{|V}}$, де $U_{|V} = \{x_v | v \in V\}$ позначає ті елементи всесвіту, які зустрічаються як мітки в G, див. Рисунок 2.1 - Графік подій на чотири вершини та пов'язаний графік оформлення. Кожен вузол графіка події замінюється чотирма вузлами, оформлені підмножинами.. Ми називаємо цей більший графік оздобленим графіком dec (G), оскільки спосіб відобразити цей вторинний графік полягає в тому, щоб кожний вузол v з G "був оформлений" з підмножинами $X \subseteq U_{|V}$. (Ми визначаємо набір вершин, використовуючи $2^{U_{|V}}$ для того, щоб дозволити кожній можливій початковій підмножині X.) Нехай n - кількість вузлів в G. Оскільки $|U_{|V}| \leq n$, ребро (v, w) в оригінальному графі породжує до 2n краю (v, X) (w, Y) у декорованому графу, причому Y виводиться з X очевидним чином. Тривіальна верхня межа числа станів $n2^n$, який є по суті жорсткою. Якщо ми могли дозволити собі зберігати весь dec (G), то будь-яка операція в вузлах графа подій може бути попередньо визначена, і час роботи на крок буде постійним. Однак потрібний простір може бути величезним, тому головне питання

Чи може декорований графік бути стиснутий без втрати продуктивності?

Здається, що складно відповісти на це питання в цілому. Фактично, навіть підрахування можливих активних наборів у оформлених графіках видається

надзвичайно нетривіальним, оскільки це зводиться до підрахунку слів у звичайних мовах, доповнених певними правилами скасування.

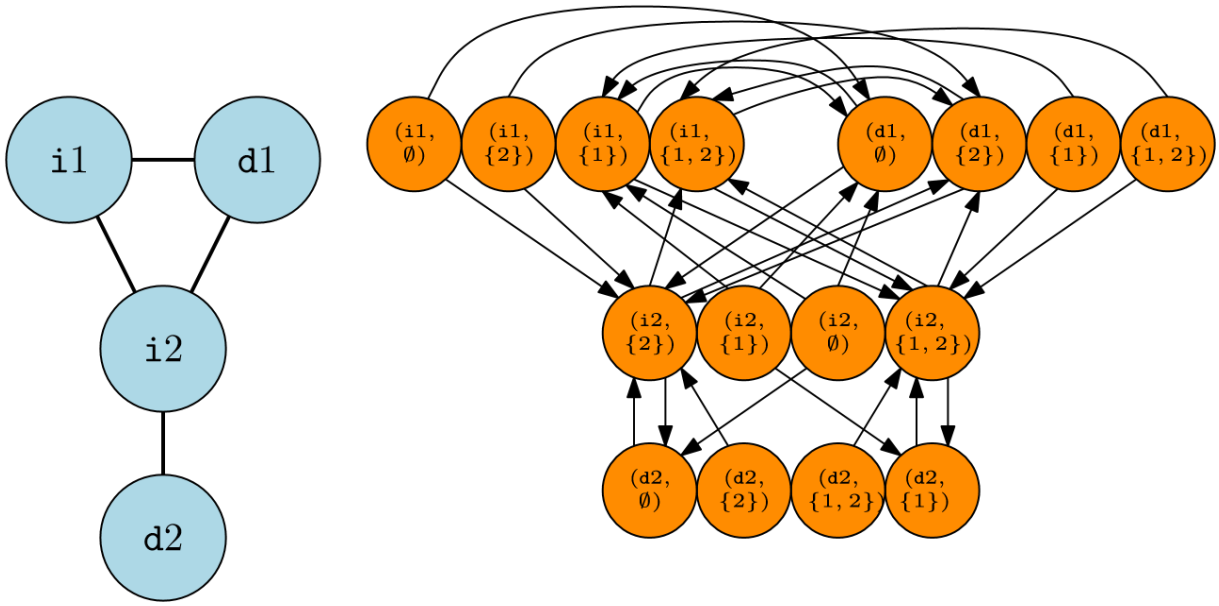

Рисунок 2.1 - Графік подій на чотири вершини та пов'язаний графік оформлення. Кожен вузол графіка події замінюється чотирма вузлами, оформлені підмножинами.

2.8 Багатозначна модель

Багатозначні бази даних є «кусковими» даними, оскільки вони можуть зберігати дані так само, як реляційні бази даних, але також дозволяють рівень глибини, який реляційна модель може наближати тільки за допомогою підтаблиць. Це майже ідентично тому, як XML висловлює дані, коли задане поле атрибут може мати кілька правильних відповідей одночасно. Багатозначність можна розглядати як стислу форму XML.

Перевага полягає в тому, що атомарность фактури (концептуального) і фактури (представлення даних) взаємно однозначна. Це також призводить до меншої кількості читань, меншої кількості проблем з посилальної цілісністю і значного зниження апаратного забезпечення, необхідного для підтримки заданого обсягу транзакцій.

2.9 Об'єктно-орієнтована модель

У 1990-х роках парадигма об'єктно-орієнтованого програмування застосовувалася до технології баз даних, створюючи нову модель бази даних, відому як бази даних об'єктів. Це направлено на запобігання невідповідності об'єктно-реляційного імпедансу - накладні витрати на перетворення інформації між її поданням до базі даних (наприклад, у вигляді рядків в таблицях) і його уявлення в прикладній програмі (як правило, як об'єкти). Більш того, система типів, використовувана в конкретному додатку, може бути визначена безпосередньо в базі даних, дозволяючи базі даних застосовувати одні і ті ж самі інваріанти цілісності даних. Бази даних об'єктів також вводять ключові ідеї об'єктного програмування, такі як інкапсуляція і поліморфізм, в світ баз даних.

Бази даних об'єктів постраждали через відсутність стандартизації: хоча стандарти були визначені ODMG, вони ніколи не були реалізовані досить добре, щоб забезпечити сумісність між продуктами. Проте, бази даних об'єктів успішно використовувалися в багатьох додатках: зазвичай спеціалізовані додатки, такі як технічні бази даних або бази даних молекулярної біології. Проте, ідеї об'єктної бази даних були сприйняті постачальниками реляційних баз даних і вплинули на розширення, зроблені для цих продуктів, і на мову SQL.

Альтернативою переходу між об'єктами і реляційними базами даних є використання бібліотеки об'єктно-реляційного зіставлення (ORM).

Висновки

Проведений аналіз існуючих моделей репрезентації частково структурованих даних у базах даних, зокрема, ієрархічної, мережевої, інвертованої файлової, реляційної, пост-реляційної, графової, багатозначної, об’єктно-орієнтованої тощо. Показано, що найбільш ефективними при вирішенні задач управління даними, які мають складну мережеву структуру, є графові моделі.

Виявлено що найбільш вживаними у рамках реалізації CMS на даний момент є реляційні бази даних, які з часом отримали деякі риси інших систем, зокрема:

- зберігання неструктурованих даних (як json поле);
- зберігання зав’язків у базі даних;
- ієрархічне представлення даних ;
- методи рекурсивного перебору дерев;
- тощо.

Якщо ж треба обробляти неструктуровані дані, то ефективність таких CMS погіршуються, із за непридатності використання існуючих методів для обробки такого типу даних.

У цьому випадку графові моделі зберігання і обробки даних мають значні переваги над реляційними при роботі з неструктурованим контентом.

3 ДОСЛІДЖЕНЯ ГРАФОВИХ МОДЕЛЕЙ ДЛЯ ОПИСУ І ОБРОБКИ ЧАСТКОВО СТРУКТУРОВАНОГО КОНТЕНТУ

3.1 Класифікація графових моделей для вирішення задач обробки неструктурованих даних.

Рубін (1976) класифікував відсутні дані у трьох категоріях: «Відсутність повністю випадково» («MCAR»), Відсутність випадково (MAR) та Відсутність не випадково (MNAR) на основі статистичних залежностей між механізмами відсутності (R-змінні) та змінними в наборі даних (V_m,V_o). Ми фіксуємо сутність цієї класифікації у графічному вигляді нижче.

a) Дані є MCAR, якщо $V_m \cup V_o \cup U \perp\!\!\!\perp R$ наявне в m-графіку. Словами, відсутність виявляється випадковим чином і цілком незалежний від спостережуваних і частково спостережуваних змінних. Цей стан можна легко ідентифікувати в m-графіку через відсутність ребр між R-змінами та змінних в $V_o \cup V_m$.

b) Дані є MAR якщо $V_m \cup U \perp\!\!\!\perp R|V_o$ наявне в m-графіку. Словами, залежність від повністю зафіксованих змінних V^o, пропуски відбуваються випадковим чином. У графічному плані MAR це, якщо (i) нема граней між змінною R та будь-якою частково спостережуваною змінною, і (ii) між змінною R і повністю спостерігаємо. змінною не існує двонаправленого ребра. MCAR означає те що і MAR, оскільки всі методи оцінки, що застосовуються до MAR, можуть бути безпечно застосовані до MCAR.

c) Дані, які не є MAR або MCAR, підпадають під категорію MNAR.

m-графіки на Рисунок 2.1 (b), (c) та (d) є типовими прикладами категорій MCAR, MAR та MNAR, відповідно. Зверніть увагу на те, якою мірою можна визначити три категорії. Коли користувач викладає взаємозв'язок між змінними у проблемі, класифікація є чисто механічною.

3.2 Дослідження графових моделей даних на основі марковських ланцюжків

Для дослідження ефективності використання графової моделі на основі марковських ланцюжків дослідимо її основні характеристики.

Наступний приклад (Little, 2002) описує як графічні моделі можуть бути використані для явного моделювання процесу відсутності даних контенту і кодування основних причинних та статистичних припущень.

Розглянемо дослідження, проведене в школі, яке вимірює три (дискретні) змінні: вік (A), стать (G) та ожиріння (O).

Без відсутності. Якщо всі три змінні повністю записані, то немає відсутностей

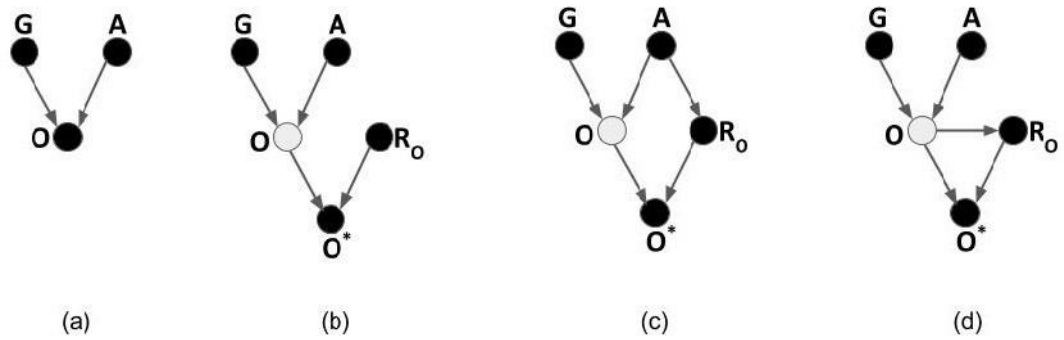

Рисунок 3.1 - (a) каузальний графік без втрат (b), (c) та (d) m-графіки, що моделюють різні процеси відсутності

Причинно-наслідкові діаграми, що відображають взаємозв'язки між змінними, показані на Рисунок 3.1 (a). Вузли відповідають змінним, а ребра вказують на наявність причинно-наслідкового зв'язку, відносини між парами вузлів, з якими вони з'єднуються. Значення дочірнього вузла є (стохастичною) функцією значень її батьківських вузлів. Наприклад, ожиріння - це (стохастична) функція віку та статті. Відсутність ребра між віком та статтю вказує на те, що A та G незалежні, позначені A⊥⊥G.

Таблиця 3.1 - Відсутність набору даних, в якому показники Вік та Стать повністю зафіксовані, і частково спостерігається ожиріння

#	Вік A	Стать G	Ожиріння O	R_O
1	16	Ж	Ожиріння	0
2	15	Ж	m	1
3	15	Ч	m	1
4	14	Ж	Немає	0
5	13	Ч	Немає	0
6	15	Ч	Ожиріння	0
7	14	Ж	Ожиріння	0

Представлення відсутності. Припустимо, що показники Вік та Стать повністю зазначені, оскільки вони можуть бути отримані з шкільних записів. Однак ожиріння невказане відсутніми значеннями через те, що деякі студенти не зважують свою вагу. Коли значення O відсутнє, ми отримуємо порожній вимір, який ми позначаємо через m. Таблиця 3.1 демонструє відсутність набору даних. Процес відсутності можна моделювати за допомогою проксі-змінної ожиріння*(O*), значення якого визначаються ожирінням та його відсутністю механізму R_O.

$$O^* = \mathrm{f}(R_0, \mathrm{O}) = \begin{cases} O \; if \; R_o = 0 \\ m \; if \; R_0 = 1 \end{cases}$$

R_O регулює маскування та розкриття ожиріння. Коли $R_O = 1$ значення ожиріння приховується, тобто O^* приймає значення m, як показано у зразках 2 та 3 у таблиці 1. Коли $R^O = 0$, виявляється справжнє значення ожиріння, тобто O^* передбачає основне значення ожиріння як показані у зразках 1, 4, 5, 6 та 7 у таблиці 1.

Відсутність може бути викликана випадковими процесами або може залежати від інших змінних в наборі даних. Прикладом випадкової відсутності є студенти, які забувають повертати свої анкети. Це зображено на Рисунок 3.1 (b) відсутністю батьківських вузлів для R_O. Підлітки, які протестують і не повідомляють про свою вагу, є прикладом відсутності, викликаного цілком спостережуваною змінною. Це зображено на Рисунок 3.1(c) краєм між A та R_O.

Частково спостережувані змінні також можуть бути причиною відсутності. Наприклад, вважають студентів з ожирінням, котрі соромляться свого ожиріння і, отже, неохоче виявляють свою вагу. Це зображено на Рисунок 3.1 (d) краєм між O та R_O що вказує на те, що O є причиною власної відсутності.

Наступний підрозділ формально вводить графіки відсутності (m-графіки), як описано в (Mohan, 2013 pp. 1277–1285).

Графіки відсутності: позначення та термінологія

Нехай G(**V**,E) причинний DAG, де V - сукупність вузлів, а E - сукупність кромки. Вузли графіка відповідають змінам у наборі даних і розділені на п'ять категорій, тобто

$$\boldsymbol{V} = V_0 \cup V_m \cup U \cup V^* \cup R$$

V_o - це сукупність змінних, які спостерігаються у всіх записах у виборці, і V_m - це сукупність змінних, що відсутні принаймні в одному записі. Змінна X називається повністю спостерігаємою, якщо $X \in V_o$ і частково спостерігаємою, якщо $X \in V_m$. Rv_i і V_i^* є двома змінними, пов'язані з кожною частково спостережуваною змінною, де V_i^* є проксі-змінною, яка фактично спостерігається, а Rv_i являє собою статус причинного механізму, що відповідає за відсутність V_i^* формально.

$$v_i^* = f(r_{v_i}, v_i) = \begin{cases} v_i & if\ r_{v_i} = 0 \\ m & if\ r_{v_i} = 1 \end{cases}$$

V^* це сукупність усіх проксі-змінних, а **R** - сукупність всіх причинних механізмів, які відповідають за відсутність. Якщо не зазначено інше, передбачається, що жодна змінна у $V_o \cup V_m \cup U$ є дітищем змінної R. U - це сукупність неспостерігаємих вузлів, що також називаються латентними змінними.

Два вузла X і Y можуть бути з'єднані прямим ребром, тобто X → Y, що вказує на те, що X є причиною Y, або двонаправленим краєм X <-> Y, що позначає існування U-змінної, яка є батьком обох змінних X та Y.

Ми називаємо це **графічне представлення графіка відсутності** (або m-графіком). На Рисунок 3.1 зображено три m-графіки, в яких $V_o = \{A,G\}$, $V_m = \{O\}$, $V^* = \{O^*\}$, $U = \emptyset$ та $R = \{RO\}$. Проксі змінні не завжди можуть бути явно відображені в m-графіках, щоб зберегти їх простими та зрозумілими. Розподіл відсутніх даних, $P(V^*,V_o,R)$ називається маніфестом розподілу та розподілом, який ми отримали б, якщо б не було пропусків, $P(V_o,V_m,R)$ називається основним розподілом. Умовні незалежності зчитуються з графіка за допомогою критерію d-поділу (Pearl, 2009). Наприклад, Рисунок 2.1(c) зображує незалежність $R_o \perp\!\!\!\perp O|A$ але не $R_O \perp\!\!\!\perp G|O$.

Похідна класифікація, використовувана в (Rubin, 1978), дуже схожа на таку, що визначена в попередніх пунктах; однак виражається з точки зору умовності на рівні подій на відміну від незалежнностей на рівні змін. Ми будемо визначим відмінність між першим (який ми називаємо Rubin-MAR), та останнім (називається MAR) за допомогою прикладу. Розглянемо набір даних з трьома змінними, такими, що A та B частково спостерігаємі змінні, а третя - C повністю спостерігається. Для того, щоб дані були MAR, ми вимагаємо щоб $(A, B) \perp\!\!\!\perp (R_A, R_B)$. З іншого боку Rubin-MAR вимагає, щоб "пропускання залежало лише від компонентів Y_{obs} з Y, які спостерігаються, а не від відсутніх компонентів" (Little, 2002), де Y позначає набір даних. Ми показуємо Rubin-MAR у таблиці 2. Основна відмінність двох визначень полягає в тому, що MAR - це стислий виклад, що складається з єдиної умовної незалежності: $V_m \perp\!\!\!\perp R \mid V_o$, у свою чергу, Rubin-MAR є сукупністю різних форм умовних незалежностей: $Y_{mis} \perp\!\!\!\perp R \mid Y_{obs}$, одна для кожної підгрупи, як описано у схемі відсутності. Зауважте, що обидва визначення збігаються, коли $| V_m | = 1$.

Протягом багатьох років класифікація, запропонована в (Rubin, 1976), піддавалася критиці як за визначення, так і за непрозорость. Кілька авторів відзначають, що MAR є некоректним ((Scheffer, 2002), (Peters, 2002), (Graham, 2009). Те, що в даний час визначено як MCAR слід назвати Missing At Random і як зазначено Грейс-Мартін, те що в даний час визначається як Missing At Random слід називати Missing Conditionally At Random.

Таблиця 3.2 - Rubin-MAR, деталізований для набору даних, в якому A та B є частково спостережуваними змінними, а C - це повністю спостережувана змінна.

Відсутні Компоненти Y_{mis}	Спостережувані Компоненти Y_{obs}	Rubin-MAR Умови $Y_{mis} \perp\!\!\!\perp R \mid Y_{obs}$	Опис зразків
A	B,C	$A \perp\!\!\!\perp R \mid B,C$	Зразки, в яких A відсутня, і B спостерігається
B	A,C	$B \perp\!\!\!\perp R \mid A,C$	Зразки, в яких відсутній B, і A спостерігається
A,B	C	$(A,B) \perp\!\!\!\perp R \mid C$	Зразки, в яких відсутні і A, і B
–	A,B,C	–	Зразки, в яких спостерігаються всі змінні.

Проте непрозорість припущень, що лежать в основі Марка Рубіна, є більш серйозною проблемою. Кількість умов умовної незалежності, що потребує перевірки, експоненціальна в кількості частково спостережуваних змінних. Це показано в Таблиця 3.2, яка відображає умовні незалежності, що вимагаються умовами Rubin-MAR: $Y_{mis} \perp\!\!\!\perp R \mid Y_{obs}$. Зрозуміло, що дослідник знайде, що він когнітивно невибагливий, якщо можливо вирішити, чи є будь-яке з цих припущень обґрунтованим. Разом з тим, Rubin-MAR є неперевіряємим, що мотивує використовувати таксономію на основі змінних, представлену вище.

Тим не менш, Rubin-MAR має цікаву теоретичну властивість: це найслабкіша проста умова, при якій процес, який призводить до відсутності, можна ігнорувати, одночасно роблячи правильні висновки щодо даних (Rubin, 1976). Ймовірно, це був теоретичний результат, який змінив практикиу в 1970-х роках. Популярна практика до 1976 р. полягала в тому, щоб припустити, що відсутність була викликана повністю випадковим чином ((Gleason, 1975), (Haitovsky, 1968)). Оскільки Рубін ідентифікує стан MAR як достатній для

наведення правильних висновків, MAR стала основним об'єктом уваги в статистичній літературі.

Процедури оцінки, такі як «Багаторічна оцінка» та «Максимальна вірогідність», були розроблені та впроваджені з урахуванням припущень MAR, а популярні підручники були розроблені винятково з використанням MAR та її спрощених версій (Graham, 2012). Ці події сформували культуру зі схильністю сліпо приймати MAR, з наслідком чого найпоширеніший клас проблем MNAR залишається відносно невивченим.

Щоб подолати ці обмеження, (Rubin, 1976) рекомендував дослідникам явно моделювати процес відсутності:

> The inescapable conclusion seems to be that when dealing with real data, the practising statistician should explicitly consider the process that causes missing data far more often than he does. However, to do so, he needs models for this process and these have not received much attention in the statistical literature.

Рисунок 3.2 - Цитата з книги Rubin (1976)

Ця рекомендація пропонує фактично графічні інструменти, описані в цьому документі, оскільки вони заохочують дослідників моделювати деталі процесу відсутності, а не сліпо приймати MAR. Ці інструменти дали змогу дослідникам розширити аналіз оцінки до великого класу проблем MNAR.

Висновки

Усі методи відсутності аналізу даних спираються на припущення щодо причин відсутності. Використання цих припущень у графічній моделі дозволяє дослідникам скористатися притаманною прозорістю таких моделей, а також їх здатністю висвітлювати статистичне втілення основних припущень з точки зору умовних відносин незалежності серед спостережуваних та частково спостережуваних змінних. Ми показали, що ці особливості графічних моделей можуть бути використані для дослідження неосвоєних територій відсутніх даних. Зокрема, ми намітили оцінку статистичних та причинних параметрів у широких класах проблем MNAR та тестування модельних припущень у умовах відсутності.

Недоліком такого підходу є його незастосовуваність для задачі управління контентом. Томи ми розробимо свій механізм роботи з частково структурованими даними.

Проведені дослідження характеристик класичних графових моделей баз даних, графових моделей на основі марковських ланцюжків та ієрархічних графових моделей для подання структур баз даних, які показали, що найбільш підходячим варіантом організації даних для даного класу задач є використання саме класичних графічних моделей.

З урахуванням результатів проведених досліджень запропоновано адаптувати класичну графову модель для представлення даних при вирішенні задач управління частково структурованим веб-контентом. Окрім того, запропонований метод підвищення структуризації контенту за рахунок створення власних типів даних, що , як наслідок, прискорює їх пошук і вибірку.

4 МЕТОД АДАПТАЦІЇ ГРАФІЧНОЇ МОДЕЛІ БАЗ ДАНИХ ДЛЯ УПРАВЛІННЯ ЧАСТКОВО СТРУКТУРОВАНИМ КОНТЕНТОМ.

Існує брак прикладів організації даних у вигляді графів. Ми розглянемо приклад організації даних в графовой БД Neo4j на прикладі моделі даних для інформаційного порталу.

4.1 Архітектура програмного рішення

Для розробки рішення було використано PHP 7 Symfony фреймворк. Він має інтегровану у себе ORM Doctrine, для абстрагування від реалізацій зберігання даних на рівні бд. Завдяки тому що білшість пакетів розширення (бандлів) для Symfony працюють з базою за допомогою ORM Doctrine ми змогли замінити базу даних з реляційної на графову без впливу на роботу стандартних бандлів (як то наприклад управління користувачами за допомогою FOS USER BUNDLE). Для ренедеренгу даних використовується шаблонізатор Twig, Він має підтримку розширень, що дозволяє нам добавити методи для виводу даних з бд на рівень оформлення.

Для уніфікації оформлення для користувачів використовується набір компонент SemanticUI. Перевикористання цих компонент дозоляє будувати інтерфейс користувача динамічно, спираючись на структури в базі даних як на метаінформацію для побудови інтерфейсу.

Щоб прискорити роботу сайтів для споживачів поперед PHP встановленно NGiNX для віддачі статичного контенту та для кешування в пам’яті побудованих сторінок. Також для пришвидшення швидкодії задіяно кеш на рівні ORM Doctrine у сховищі Redis.

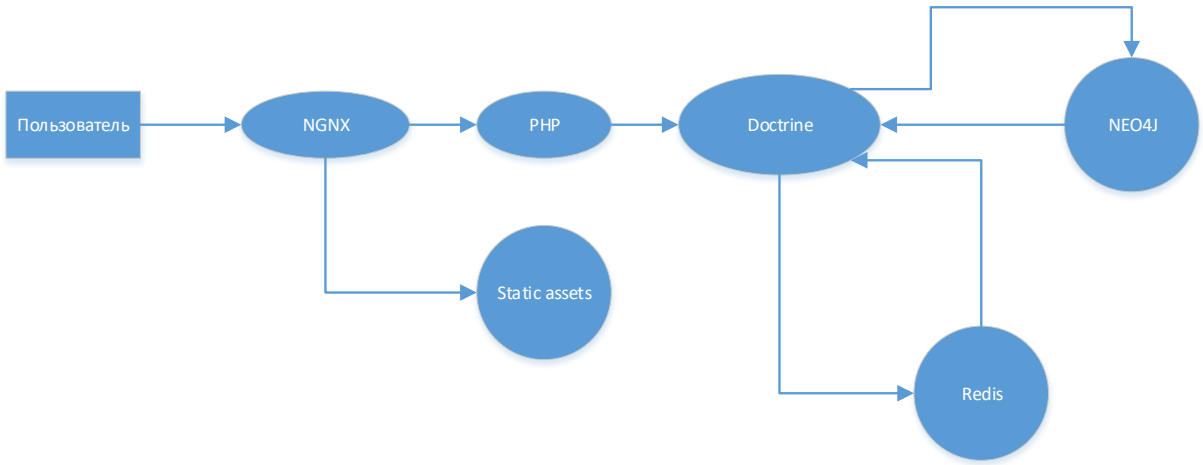

Рисунок 4.1 - ілюстрація компонентів программи та їх зв'язок при обробці запросу від пользователя. (Кожен елемент виконується в окремому докер контейнері)

4.2 Структура даних у базі даних

На відміну від реляційної БД зв'язку в графовой базі можуть мати типи, атрибути і значення. Таким чином є дві основні сутності - вузол і зв'язок.

Розберемо приклад новинного порталу. Типи даних які будуть в ньому використовуватися: Стаття, Автор, Новина, Огляд.

Тип даних - це схема, що описує повторювану сторінку. Наприклад, всі статті мають поля "анотація", "заголовок", "дата публікації". Тобто тип даних, це метаописів всіх об'єктів даного типу, в якому зберігається список полів і інформація про типи полів (типи полів можуть бути комбінованими або структурними, наприклад, список допустимих значень).

Тип даних пов'язаний з екземплярами даних. Екземпляр це кінцева стаття. У неї є значення, які комплементарні опису типу. Логіка роботи з типами даних наведена у Додаток А.

Крім того в описі типу може бути вказаний зв'язок з іншим типом. Для Статті це може бути зв'язок з Автором. У кожного екземпляра статті буде двонаправлений зв'язок з автором. Це означає, що у кожного екземпляра автора будуть зв'язки з усіма його статтями.

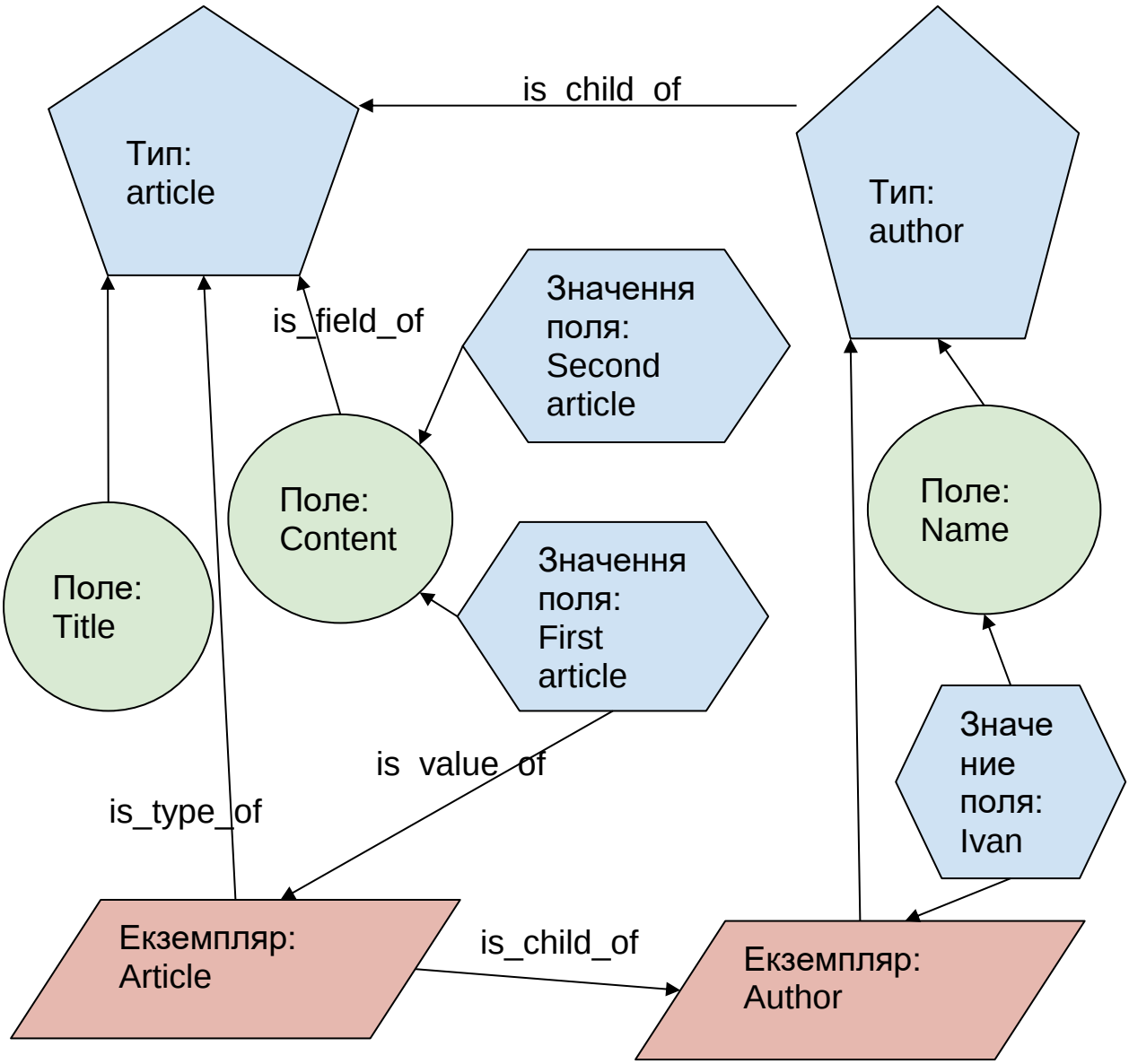

Рисунок 4.2 - Приклад схеми зав'язків у графовій базі данних

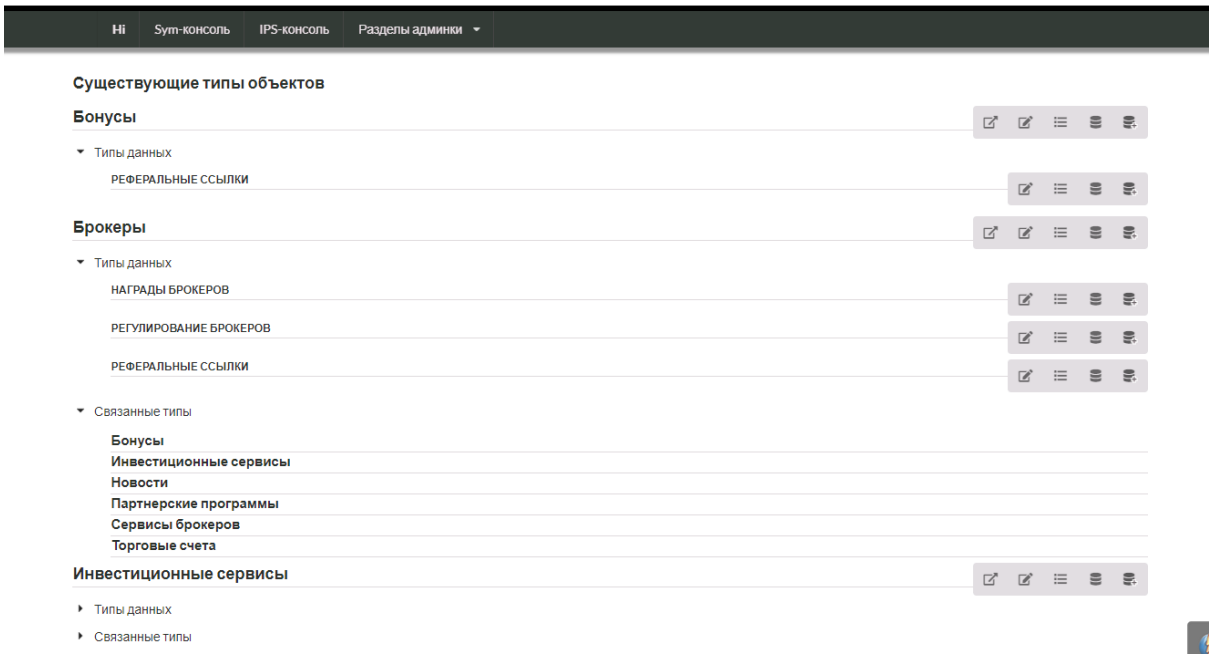

Рисунок 4.3 - приклад адмін панелі для редагування типів данних редактором

Наведемо декілька прикладів работи з цими даними за допомогою мови запросів Cypher.

Пошук вісх полей та значень обьекту:

```
MATCH (etf:EntityTypeField)-[:has_field]->(et:EntityType)<-[:has_type]-(o:Object)
  WHERE id(o) = {oId}
  OPTIONAL MATCH (o)<-[rel:is_field_of]-(fv:FieldValue)-[:is_value_of]->(etf)
  WITH rel, etf, fv ORDER BY etf.order, rel.order
  WITH etf, collect(fv) as val
  RETURN etf as type, val
```

Пошук конкретного предка за псевдонімом:

```
MATCH (pet:EntityType)<-[:is_child_of]-(et:EntityType)<-[:has_type]-(o:Object)
        WHERE id(o) = {oId} AND pet.slug = {slug}
        RETURN pet LIMIT 1
```

Видаленя значень поля:

```
$removeRelationQuery = 'MATCH (o2:Object)<-[:is_field_of]-(fv:FieldValue)-[obj_field_rel:is_field_of]-
>(o:Object)-[:has_type]->(et:EntityType)
            <-[:has_field]-(etf:EntityTypeField)-[etfr:is_value_of]-(fv)
        WHERE id(o) = {oId} AND (o <> o2) AND id(etf)= {etfId} AND id(fv) in {ids}
        DELETE obj_field_rel';
    $detachRemoveQuery = 'MATCH (fv:FieldValue)-[:is_field_of]->(o:Object)-[:has_type]-
>(et:EntityType)
            <-[:has_field]-(etf:EntityTypeField)-[etfr:is_value_of]-(fv)
        WHERE id(o) = {oId} AND id(etf)= {etfId} AND id(fv) in {ids}
        DETACH DELETE fv';
```

Можна побачити деяку схожість з SQL, але модифікований для роботи з графами, для цього введені модифікатори полів, що менш наглядно аніж JOINи в SQL, але позволяэ набагато компактніше виразити завдання на вібірку.

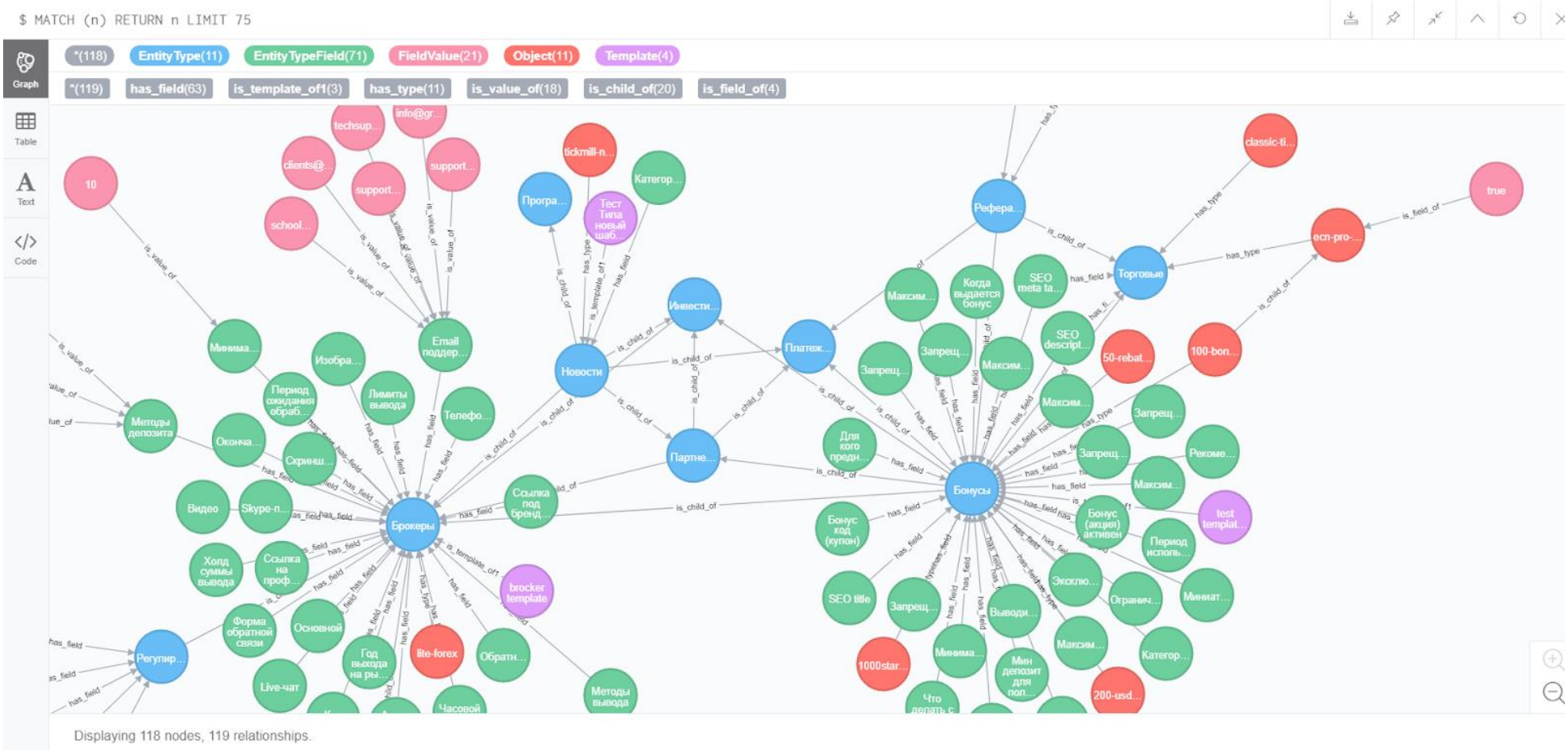

Рисунок 4.4 - приклад проекції даних на графову базу даних

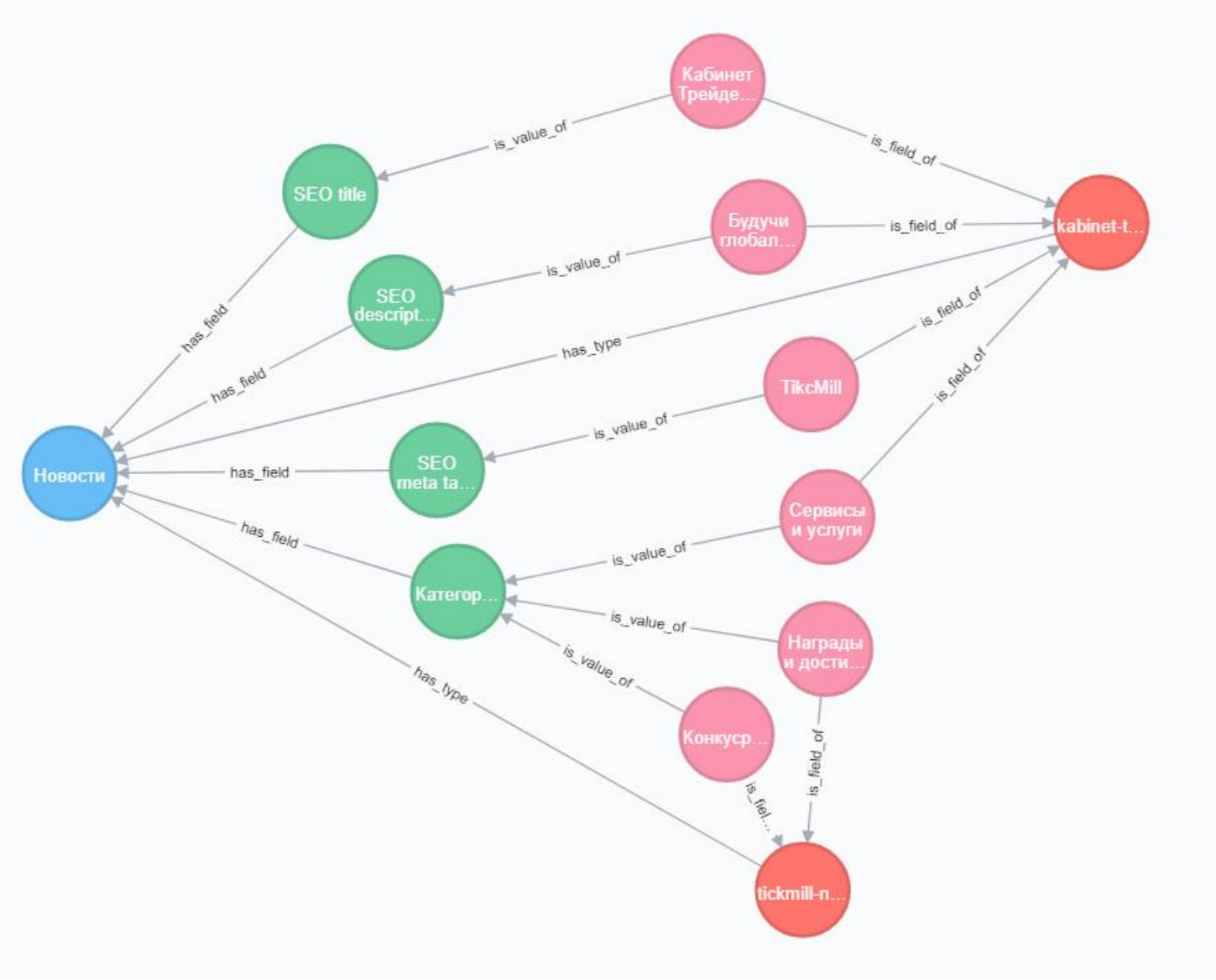

Рисунок 4.5 - тип даних "Новини" ілюструє структуру зв'язків "значення поля-об'єкт-тип-тип поля-значення поля"

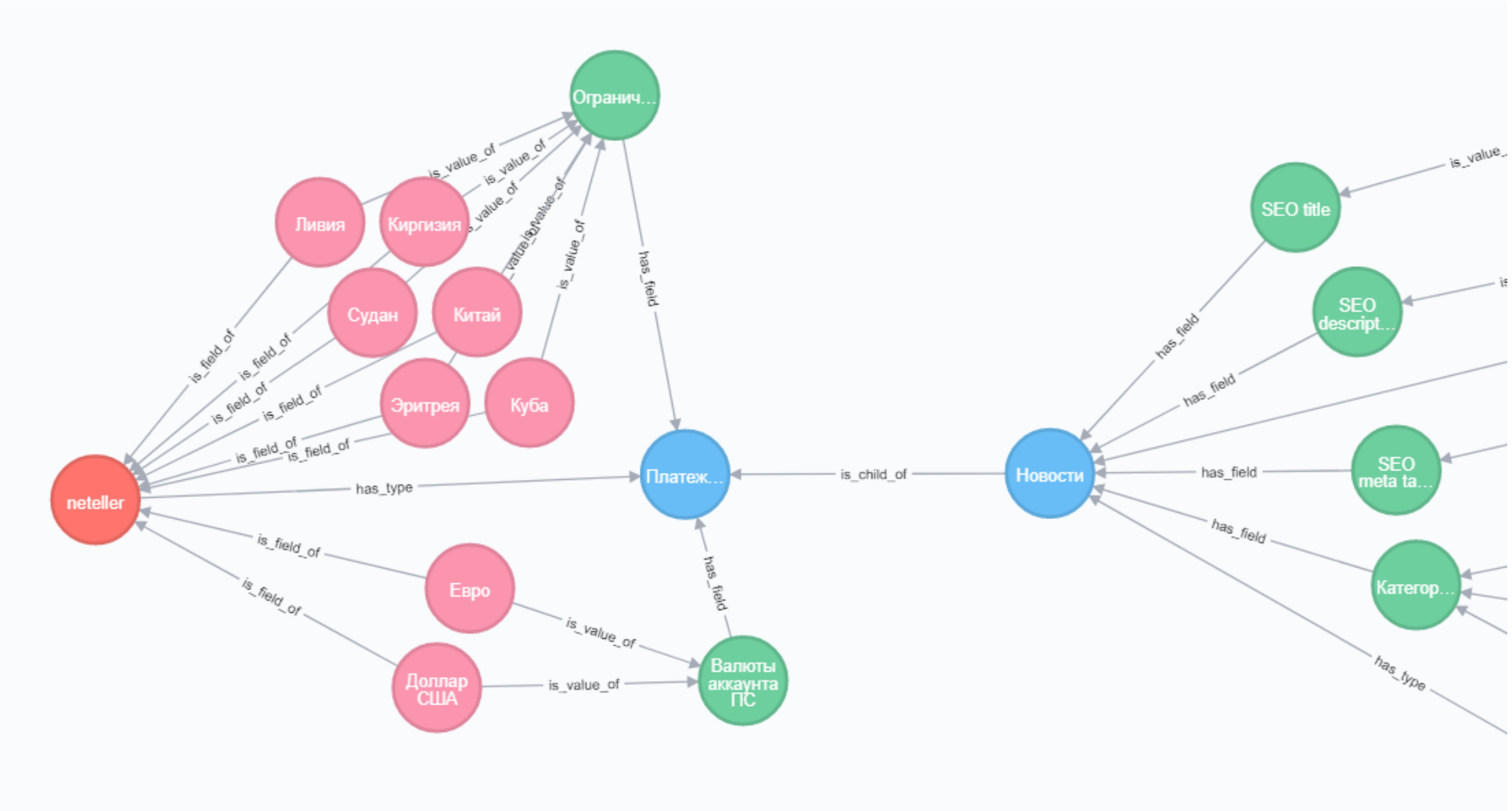

Рисунок 4.6 - Ілюстрація зв'язку між типами «Платіжні системі» та «Новини». Наявність цього зв'язку дозволяє мати в системі «новина о платіжній системі»

Щоб оперувати зв’язками міх типами ми вводимо поняття зв’язних типів. Для внесення зв’язку такого типу необхідно задекларувати тип об'єкта на який можна посилатися в схемі опису типу. Таким чином валідація таких зв'язків при створенні досить трудомістка, але дуже легко виводити все пов'язані об'єкти в статтях при рендерингу сторінки. Ми просто говоримо - вивести всі вузли з типом зв'язку не далі ніж в двох стрибках від нас. У WP немає аналогів, тому що там підтримується пряме лінкування статей - грубо кажучи хардкод зв'язків в конетнт. До версії 3 був менеджер посилань в WP, але далі його підтримка в рамках парадигми WP виявилася надто витратною і цей функціонал прибрали).

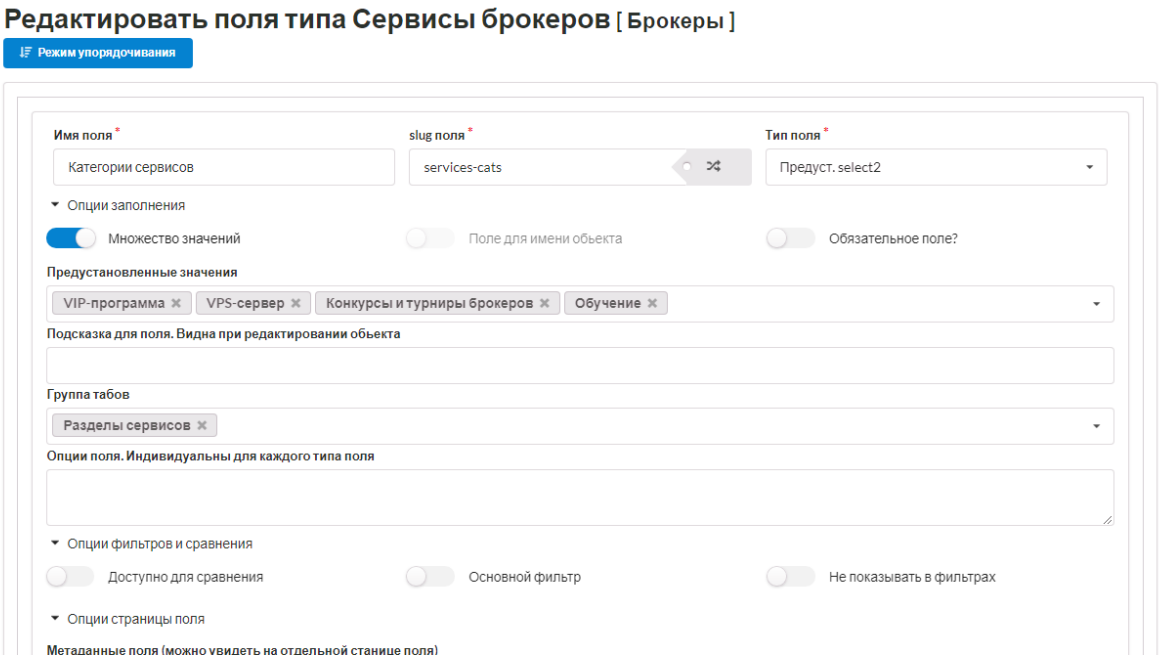

Рисунок 4.7 - Приклад редактора поля у типу

Висновки

Запропоновано метод адаптації класичної графової моделі бази даних для управління частково структурованим веб контентом.

Запропонований метод створення власних типів даних контенту, що відповідають обраній структурі бази даних

На основі запропонованих методів розроблено архітектуру CMS на прикладі інформаційного порталу, яка дає можливість використання більш структурованого контенту.

Побудовано конструктор типів контенту для CMS, шо грунтується на використанні запропонованої графової моделі, та розроблені відповідні шаблони для візуалізації даних, внесених користувачами порталу.

З урахуванням запропонованих рішень розроблена система управління контентом, на базі якої були імплементовані наступні проекти - https://coin.ua, https://homeinvesting.ru.

Наукова новизна отриманих результатів полягає у розробці та реалізації методу структуризації даних, які мають складну мережеву структуру, за допомогою використання графових баз даних і динамічного створення власних типів даних, що підвищує структуризацію контенту і, як наслідок, прискорює пошук і вибірку даних.

ВИСНОВКИ

У даній магістерській дисертації запропонований та адаптований в рамках розробленої системи управління контентом метод структуризації даних на основі використання графових баз даних для підвищення продуктивності роботи систем управління контентом при роботі з даними, які мають складну мережеву структуру.

Виконаний аналіз існуючих систем управління контентом з точки зору реалізації структур баз даних ї задач управління великими обсягами частково структурованого веб-контенту. Показано, що із зростанням кількості статей реляційна модель організації даних, загальноуживана у сучасних CMS, зокрема, у WordPress, призводить до уповільнення операцій пошуку і виборки по даним. Шляхом вирішення даної проблеми може бути використання іншої моделі організації даних, зокрема, уникнення необхідності побудови JOIN-ів для реалізації зв’язків між типами даних і самими даними.

Проведений аналіз існуючих моделей репрезентації частково структурованих даних у базах даних, зокрема, ієрархічної, мережевої, інвертованої файлової, реляційної, пост-реляційної, графової, багатозначної, об’єктно-орієнтованої тощо. Показано, що найбільш ефективними при

вирішенні задач управління даними, які мають складну мережеву структуру, є графові моделі.

Проведені дослідження характеристик класичних графових моделей баз даних, графових моделей на основі марковських ланцюжків та ієрархічних графових моделей для подання структур баз даних, які показали, що найбільш підходячим варіантом організації даних для даного класу задач є використання саме класичних графічних моделей.

З урахуванням результатів проведених досліджень запропоновано адаптувати класичну графову модель для представлення даних при вирішенні задач управління частково структурованим веб-контентом. Окрім того, запропонований метод підвищення структуризації контенту за рахунок створення власних типів даних, що , як наслідок, прискорює їх пошук і вибірку.

На основі запропонованих методів розроблено архітектуру CMS на прикладі інформаційного порталу, яка дає можливість використання більш структурованого контенту.

Побудовано конструктор типів контенту для CMS, шо грунтується на використанні запропонованої графової моделі, та розроблені відповідні шаблони для візуалізації даних, внесених користувачами порталу.

З урахуванням запропонованих рішень розроблена система управління контентом, на базі якої були імплементовані наступні проекти - https://coin.ua, https://homeinvesting.ru.

Наукова новизна отриманих результатів полягає у розробці та реалізації методу структуризації даних, які мають складну мережеву структуру, за допомогою використання графових баз даних і динамічного створення власних типів даних, що підвищує структуризацію контенту і, як наслідок, прискорює пошук і вибірку даних.

Таким чином, усі поставленні завдання даної магістерської роботи виконані, а поставлена мета – досягнута. Використання запропонованого методу структуризації даних підвищує продуктивність роботи систем управління

контентом при вирішенні задач управління великими об’ємами даних, які мають складну мережеву структуру.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) **Chazelle B.** The discrepancy method: randomness and complexity [Книга]. - Cambridge : Cambridge University Press, 2000.
2) **D. A. Levin Y. Peres, and E. L. Wilmer** Markov chains and mixing times [Книга]. - Providence : American Mathematical Society, 2009.
3) **Daniel R. M., M. G. Kenward, S. N. Cousens, and B. L. De Stavola** Using causal diagrams to guide analysis in missing data problems [Стаття] // Statistical methods in medical research. - 2012 р.. - 23 (3).
4) **Darwiche A** Modeling and reasoning with Bayesian networks [Article] // Cambridge University Press. - Cambridge : [s.n.], 2009.
5) **Dempster A., N. Laird, and D. Rubin** Maximum likelihood from incomplete data via the em algorithm [Книга]. - [місце видання невідоме] : Journal of the Royal Statistical Society, 1977. - T. Series B (Methodological).
6) **Enders C** Applied Missing Data Analysis [Book]. - [s.l.] : Guilford Press, 2010.
7) **Gleason T. C. and R. Staelin** A proposal for handling missing data. [Article] // Psychometrika. - 1975. - 40 : Vol. 2. - pp. 229-252.
8) **Graham J.** Missing Data: Analysis and Design (Statistics for Social and Behavioral Sciences). [Книга]. - [місце видання невідоме] : Springer., 2012.
9) **Graham J. W.** Missing data analysis: Making it work in the real world. [Стаття] // Annual review of psychology. - 2009 р.. - 60. - сс. 549-576.

10) **GUTIERREZ RENZO ANGLES and CLAUDIO** Survey of Graph Database Models [Article] // ACM Computing Surveys, Vol. 40, No. 1, Article 1. - February 2008. - 1.

11) **Haitovsky Y.** Missing data in regression analysis. [Journal]. - [s.l.] : Journal of the Royal Statistical Society. Series B (Methodological), 1968. - pp. 67-82.

12) **Hoff Todd** Paper: Graph Databases And The Future Of Large-Scale Knowledge Management [Online] // highscalability. - 06 10, 2009. - http://highscalability.com/paper-graph-databases-and-future-large-scale-knowledge-management.

13) http://techportal.ibuildings.com/2009/09/07/graphs-in-the-database-sql-meets-social-networks/ [Онлайновий].

14) **Koller D. and N. Friedman** Probabilistic graphical models: principles and techniques [Книга]. - 2009.

15) **Little R. and D. Rubin** Statistical analysis with missing data [Book]. - [s.l.] : Wiley, 2002.

16) **Mohan K., J. Pearl, and J. Tian** Graphical models for inference with missing data [Стаття] // Advances in Neural Information Processing Systems. - 2013 p.. - 26.

17) **Pearl J.** Causality: models, reasoning and inference. [Book]. - New-York : Cambridge Univ Press, 2009.

18) **Peters C. L. O. and C. Enders** A primer for the estimation of structural equation models in the presence of missing data: Maximum likelihood algorithms. [Стаття] // Journal of Targeting, Measurement and Analysis for Marketing. - 2002 p.. - 11 (1). - сс. 81-95.

19) **Robinson I. and Webber, J. and Eifrem, E.** [Book Section] // Graph Databases. - [s.l.] : O'Reilly Media, 2013. - ISBN 978-1449356262.

20) **Rubin D. B.** Multiple imputations in sample surveys-a phenomenological bayesian [Стаття] // Proceedings of the survey

research methods section of the. - [місце видання невідоме] : American Statistical Association, 1978 p.. - T. 1.

21) **Rubin D.B.** Inference and missing data [Книга]. - 1976.

22) **Scheffer J.** Dealing with missing data. [Стаття] // Research Letters in the Information and Mathematical Sciences. - 2002 p.. - cc. 153-160.

23) **Thoemmes F. and K. Mohan** Graphical representation of missing data problems. [Article] // Structural Equation Modeling: A Multidisciplinary Journal.. - 2015.

24) **Thoemmes F. and N. Rose** Selection of auxiliary variables in missing data problems: Not all auxiliary variables are created equal. Technical Report R-002 [Звіт]. - [місце видання невідоме] : Cornell University, 2013.

ДОДАТОК А: ДІАГРАМА ДІЯЛЬНОСТІ